175

select FirstName,LastName,City,State

from Person p

left join Address a

on p.PersonId=a.Personid

176

select(

select distinct Salary

from Employee

order by Salary desc limit 1 offset 1) as SecondHighestSalary

select distinct salary from Employee order by salary desc limit 1,1

就可以，但是输不出null，所以外面再加一层

select (select distinct salary from Employee order by salary desc limit 1,1) as SecondHighestSalary

177

CREATE FUNCTION getNthHighestSalary(N INT) RETURNS INT

BEGIN

    SET N=N-1;

    RETURN (

        select distinct Salary From Employee order by Salary desc limit 1 offset N

  );

END

178

select Score, dense\_rank()

over(order by Score desc) as 'Rank'

from Scores

180

select distinct l1.Num as ConsecutiveNums

from Logs l1

join Logs l2 on l1.Num=l2.Num and l2.id=l1.id+1

join Logs l3 on l2.Num=l3.Num and l3.id=l2.id+1

181

select e1.Name as Employee

from Employee e1

join Employee e2

on e1.ManagerId=e2.Id

where e1.Salary>e2.Salary

182

select Email

from Person

group by Email

having count(\*)>1

183

select Name as Customers

from Customers

where id not in (select CustomerId from Orders)

184

select d.Name as Department,e.Name as Employee, Salary

from Employee e

join Department d

on e.DepartmentId=d.Id

where (DepartmentId,Salary) in

(select DepartmentId,max(Salary)

from Employee e

group by DepartmentId)

185

select d.Name as Department,e.Name as Employee,Salary

from Employee e

join Department d

on e.DepartmentId=d.Id

where (DepartmentId,Salary) in(

select DepartmentId,Salary

from(

select DepartmentId,Salary,dense\_rank() over(partition by DepartmentId order by Salary desc) rnk

from Employee) t1

where rnk<=3)

196

delete p2

from Person p1

join Person p2

on p1.Email=p2.Email and p1.Id<p2.Id

197

select w2.Id

from Weather w1

join Weather w2

on datediff(w2.RecordDate,w1.RecordDate)=1

where w2.Temperature>w1.Temperature

262

select Request\_at as day, round(sum(case when Status like "%cancelled%" then 1 else 0 end)/count(\*),2) as "Cancellation Rate"

from Trips

where Client\_id not in (select Users\_id from Users where Banned="Yes") and Request\_at between '2013-10-01' and '2013-10-03'

group by Request\_at

511

select player\_id,min(event\_date) as first\_login

from Activity

group by player\_id

512

select player\_id,device\_id

from Activity

where (player\_id,event\_date) in (select player\_id, min(event\_date) from Activity group by player\_id)

534

select player\_id, event\_date,

sum(games\_played) over(partition by player\_id order by event\_date) as games\_played\_so\_far

from Activity

550

select round((

select count(distinct a1.player\_id)

from activity a1

join activity a2

on datediff(a2.event\_date,a1.event\_date)=1 and a1.player\_id=a2.player\_id

where (a1.player\_id,a1.event\_date) in (select player\_id, min(event\_date) from activity group by player\_id))/count(distinct player\_id),2) as fraction

from activity

569

select Id,Company,Salary from

(

select Id,Company,Salary,

row\_number() over(partition by Company order by Salary) as rnk,

count(Salary) over(partition by Company) as cnt from Employee

) t

where rnk in (cnt/2,cnt/2+1,cnt/2+0.5)

找出row\_number和count的关系

570

select e1.Name

from Employee e1

join Employee e2

on e1.Id=e2.ManagerId

group by e1.Id

having count(e2.Id) >=5

571

select avg(number) as median

from(

select \*, sum(frequency) over(order by Number) s1, sum(frequency) over(order by Number desc) s2

from numbers) d

where s1>=(select sum(frequency) from numbers)/2 and s2>=(select sum(frequency) from numbers)/2

频数正数，倒数都大于sum（frequency）的一半，再取对应数字的avg

574

select Name

from Candidate

where id = (select CandidateId from Vote group by CandidateId order by count(id) desc limit 1)

577

select e.name,bonus

from Employee e

left join Bonus b on e.empId=b.empId

where bonus <1000 or bonus is null

578

select question\_id as survey\_log

from(

select question\_id, sum(case when action="answer" then 1 else 0 end)/sum(case when action="show" then 1 else 0 end) as rate

from survey\_log

group by question\_id) t

order by rate desc limit 1

579

select Id, Month,

        sum(Salary) over(partition by Id order by Month **rows 2 preceding**) Salary

from Employee

where (Id, Month) not in

(

    select Id, max(Month)

    from Employee

    group by Id

)

order by Id, Month desc

580

select d.dept\_name, ifnull(count(student\_id),0) as student\_number

from department d

left join student s

on d.dept\_id=s.dept\_id

group by d.dept\_id

order by student\_number desc, dept\_name

584

select name

from customer

where referee\_id <> 2 or referee\_id is null

585

select sum(TIV\_2016) as TIV\_2016

from insurance

where TIV\_2015 in (select TIV\_2015 from insurance group by TIV\_2015 having count(\*)>1)

and concat(LAT,LON) not in (select concat(LAT,LON) from insurance group by concat(LAT,LON) having count(\*) >1)

586

select customer\_number

from orders

group by customer\_number

order by count(order\_number) desc limit 1

595

select name,population,area

from world

where population>25000000 or area>3000000

596

select class

from courses

group by class

having count(distinct student) >= 5

597

# Write your MySQL query statement below

select round(ifnull((select count(\*)

from(

select distinct requester\_id,accepter\_id

from RequestAccepted) t )/(select count(\*) from (select distinct sender\_id,send\_to\_id from FriendRequest) t2),0),2) as accept\_rate

601

select \*

from Stadium

where id in (

select t.id1

from (

select s1.id as id1, s2.id as id2, s3.id as id3

from Stadium s1

join Stadium s2 on s1.id=s2.id-1 and s1.people>=100 and s2.people>=100

join Stadium s3 on s2.id=s3.id-1 and s2.people>=100 and s3.people>=100) t

union

select t.id2

from (

select s1.id as id1, s2.id as id2, s3.id as id3

from Stadium s1

join Stadium s2 on s1.id=s2.id-1 and s1.people>=100 and s2.people>=100

join Stadium s3 on s2.id=s3.id-1 and s2.people>=100 and s3.people>=100) t

union

select t.id3

from (

select s1.id as id1, s2.id as id2, s3.id as id3

from Stadium s1

join Stadium s2 on s1.id=s2.id-1 and s1.people>=100 and s2.people>=100

join Stadium s3 on s2.id=s3.id-1 and s2.people>=100 and s3.people>=100) t)

602

select requester\_id as id, count(accepter\_id) as num

from(

select requester\_id,accepter\_id from request\_accepted

union all

select accepter\_id,requester\_id from request\_accepted) t

group by requester\_id

order by count(accepter\_id) desc limit 1

**603**

select distinct c1.seat\_id

from cinema c1

join cinema c2

on c1.free=1 and c2.free=1 and (c1.seat\_id=c2.seat\_id+1 or c1.seat\_id=c2.seat\_id-1)

607

select name

from salesperson

where sales\_id not in (select sales\_id from orders o join company c on o.com\_id=c.com\_id where c.name="RED")

608

select id,if(isnull(p\_id),"Root",if(id in (select p\_id from tree),"Inner","Leaf")) as Type

from tree

610

select x,y,z,if(x+y>z and x+z>y and y+z>x,"Yes","No") as triangle

from triangle

614

select distinct f1.followee as follower, count(distinct f1.follower) as num

from follow f1

join follow f2

on f1.followee=f2.follower

group by f1.followee

615

select d1.pay\_month,d2.department\_id,

    case

    when dept\_avg>company\_avg then "higher"

    when dept\_avg<company\_avg then "lower"

    else "same"

    end as comparison

from(

select date\_format(pay\_date,"%Y-%m") as pay\_month,avg(amount) as company\_avg

from salary

group by date\_format(pay\_date,"%Y-%m")) d1

join(

select department\_id,date\_format(pay\_date,"%Y-%m") as pay\_month,avg(amount) as dept\_avg

from salary s

join employee e on s.employee\_id=e.employee\_id

group by date\_format(pay\_date,"%Y-%m"),department\_id) d2

on d1.pay\_month=d2.pay\_month

**618**

max()函数的功效：（‘jack’, null, null）中得出‘jack'，（null,null）中得出null。 min()函数也可以。

select

max(if(continent='America',name,null)) America,

max(if(continent='Asia',name,null)) Asia,

max(if(continent='Europe',name,null)) Europe

from

(select \*, row\_number() over (partition by continent order by name) rk from student) t

group by rk;

619

select max(num) as num

from my\_numbers

where num in (select num from my\_numbers group by num having count(\*)=1)

620

select \*

from cinema

where description not like '%boring%' and id mod 2 =1

order by rating desc

626

select (case

            when id%2=1 and id=(select count(id) from seat) then id

            when id%2=1 then id+1

            else id-1

        end) as id

    ,student

from seat

order by id

627

update salary

set sex=if(sex="m","f","m")

1045

select customer\_id

from Customer

group by customer\_id

having count(distinct product\_key) = (select count(distinct product\_key) from product)

1050

select actor\_id,director\_id

from ActorDirector

group by concat(actor\_id,'-',director\_id)

having count(timestamp) >=3

1068

select product\_name,year,price

from sales s

join product p on s.product\_id=p.product\_id

1069

select product\_id,sum(quantity) as total\_quantity

from sales

group by product\_id

1070

select product\_id,year as first\_year,quantity,price

from sales

where (product\_id,year) in (select product\_id,min(year) from sales group by product\_id)

1075

select project\_id,round(avg(experience\_years),2) as average\_years

from Project p

join Employee e on p.employee\_id=e.employee\_id

group by project\_id

1076

select project\_id

from Project p

group by project\_id

having count(\*) = (select count(\*) from Project group by project\_id order by count(\*) desc limit 1)

1077

select p.project\_id,p.employee\_id

from project p

join employee e on e.employee\_id=p.employee\_id

where (p.project\_id,e.experience\_years) in

(

select project\_id,max(experience\_years)

from employee e

join project p on e.employee\_id=p.employee\_id

group by project\_id)

OR

select project\_id,e.employee\_id,

dense\_rank() over(partition by project\_id order by experience\_years desc) as rnk

from project p

join employee e on p.employee\_id=e.employee\_id

where rnk=1

1082

select seller\_id

from sales

group by seller\_id

having sum(price) = (select sum(price) from sales group by seller\_id order by sum(price) desc limit 1)

1083

select buyer\_id

from sales s

join product p on s.product\_id=p.product\_id

group by buyer\_id

having sum(case when product\_name="S8" then 1 else 0 end)>0 and sum(case when product\_name="iPhone" then 1 else 0 end)=0

OR

select buyer\_id

from (

select buyer\_id,sum(case when product\_name="S8" then 1 else 0 end) as c1, sum(case when product\_name="IPhone" then 1 else 0 end) as c2

from product p

join sales s

on p.product\_id=s.product\_id

group by buyer\_id) t

where c1>=1 and c2=0

1084

select product\_id,product\_name

from Product

where product\_id in (

select product\_id

from Sales

group by product\_id

having min(sale\_date) >= "2019-01-01" and max(sale\_date) <= "2019-03-31" )

1097

select first\_date as install\_dt, count(\*) as installs,

round(count(a.event\_date)/count(\*),2) as Day1\_retention

from (

select player\_id,min(event\_date) as first\_date

from Activity

group by player\_id) t

left join Activity a

on datediff(a.event\_date,t.first\_date)=1 and a.player\_id=t.player\_id

group by first\_date

**OR**:

select first\_day as install\_dt, count(player\_id) as installs, round(sum(case when t.sec\_day is null then 0 else 1 end)/count(player\_id),2) as Day1\_retention

from(

select a1.player\_id,a1.event\_date as first\_day,a2.event\_date as sec\_day

from activity a1

left join activity a2

on a1.player\_id=a2.player\_id and datediff(a2.event\_date,a1.event\_date)=1

where (a1.player\_id, a1.event\_date) in (select player\_id, min(event\_date) from activity group by player\_id)) t

group by first\_day

1098

select book\_id,name

from Books

where book\_id in(

select b.book\_id

from Books b

left join (select \* from Orders where dispatch\_date between '2018-06-23' and '2019-06-23') o

on b.book\_id=o.book\_id

group by b.book\_id

having ifnull(sum(quantity),0)<10) and book\_id  in (select book\_id from books where available\_from<='2019-05-23')

主要查询每本书的销量的时候要用left join而不是简单聚合Order表（Orders表Id不全）

1107

select login\_date,count(user\_id) as user\_count

from(

select distinct user\_id,min(activity\_date) as login\_date

from traffic

where activity="login"

group by user\_id

having datediff('2019-06-30',min(activity\_date)) <=90) t

group by login\_date

order by login\_date

1112

select student\_id,course\_id,grade

from (

select student\_id, course\_id,grade, row\_number() over(partition by student\_id order by grade desc,course\_id) as rnk

from enrollments) t

where rnk=1

1113

select extra as report\_reason, count(\*) as report\_count

from(

select distinct post\_id,extra

from actions

where action="report" and action\_date='2019-07-04') t

group by extra

1126

select business\_id

from(

select business\_id,e.event\_type,occurences,avg\_time,

sum(case when occurences>avg\_time then 1 else 0 end) as times

from events e

join (select event\_type,avg(occurences) as avg\_time from events group by event\_type) t

on e.event\_type=t.event\_type

group by business\_id) t1

where times>=2

1127

select t3.spend\_date,t3.platform,ifnull(sum(total\_amount),0) as total\_amount,ifnull(sum(total\_users),0) as total\_users

from(

select t.spend\_date, t.platform,total\_amount,total\_users

from (

    select distinct spend\_date,"desktop" as platform from spending

union

select distinct spend\_date,"mobile" as platform from spending

union

select distinct spend\_date,"both" as platform from spending

) t

left join(

select spend\_date,

  if(count(distinct platform)=1,platform,"both") as plat,

  sum(amount) as total\_amount,

  count(distinct user\_id) total\_users

from spending

group by spend\_date,user\_id) t2

on t.platform=t2.plat and t.spend\_date=t2.spend\_date) t3

group by spend\_date,platform

1132

from (

select a.action\_date,count(distinct r.post\_id)/count(distinct a.post\_id) as rate

from (select \* from actions where action="report" and extra="spam") a

left join removals r

on a.post\_id=r.post\_id

group by a.action\_date) t

1141

select activity\_date as day, count(distinct user\_id) as active\_users

from Activity

where datediff('2019-07-27',activity\_date)<30

group by activity\_date

最近30天是>不上>=

1142

select ifnull(round(sum(sessions)/count(user\_id),2),0) as average\_sessions\_per\_user

from (select distinct user\_id, count(distinct session\_id) as sessions

from Activity

where datediff('2019-07-27',activity\_date)<30

group by user\_id) t

1148

select distinct author\_id as id

from Views

where author\_id=viewer\_id

order by id

1149

select distinct viewer\_id as id

from Views

group by viewer\_id,view\_date

having count(distinct article\_id)>=2

order by id

1158

select user\_id as buyer\_id,join\_date,count(order\_id) as orders\_in\_2019

from users u

left join orders o

on u.user\_id=o.buyer\_id and year(order\_date)=2019

group by user\_id

1159

select user\_id as seller\_id,case when item\_brand=favorite\_brand then "yes" else "no" end as 2nd\_item\_fav\_brand

from users u

left join(

    select \*

    from (

    select \*,row\_number() over(partition by seller\_id order by order\_date) as rnk

from orders) t

    where rnk=2) t1

on u.user\_id=t1.seller\_id

left join items i on i.item\_id=t1.item\_id

1164

select t1.product\_id,case when change\_date is null then orig else new\_price end as price

from (

select distinct product\_id,10 as orig, '2000-01-01' as orig\_date

from products) t1

left join(

select product\_id,new\_price,change\_date

from (

select \*, row\_number() over(partition by product\_id order by change\_date desc) as rnk

from products

where change\_date <= '2019-08-16') t

where rnk=1) t2

on t1.product\_id=t2.product\_id

OR

select p.product\_id,case when change\_date>orig\_date then new\_price else orig\_price end as price

from (

select distinct product\_id, 10 as orig\_price,'2000-01-01' as orig\_date

from products) p

left join(

select \*

from products

where (product\_id,change\_date) in (

select product\_id,max(change\_date)

from products

where change\_date <= '2019-08-16'

group by product\_id)) t

on p.product\_id=t.product\_id

先获取指定日期内latest价格（用subqueries），再构造一个新的表个让每个品种的价格都是10，设置日期为一个相对较小的日期。在用left join如果change date大于初始日期则价格为最新的价格，否则价格为10

1173

select round(sum(case when order\_date=customer\_pref\_delivery\_date then 1 else 0 end)/count(\*)\*100,2) as immediate\_percentage

from delivery

1174

select

round((select count(\*)

from (

select \*

from delivery

where (customer\_id,order\_date) in (

select customer\_id,min(order\_date) as first\_deliv

from delivery

group by customer\_id)) t

where t.order\_date=customer\_pref\_delivery\_date)/count(distinct customer\_id)\*100,2) as immediate\_percentage

from delivery

1179

select id,

    sum(case when month="Jan" then revenue else null end) as Jan\_Revenue,

    sum(case when month="Feb" then revenue else null end) as Feb\_Revenue,

    sum(case when month="Mar" then revenue else null end) as Mar\_Revenue,

    sum(case when month="Apr" then revenue else null end) as Apr\_Revenue,

    sum(case when month="May" then revenue else null end) as May\_Revenue,

    sum(case when month="Jun" then revenue else null end) as Jun\_Revenue,

    sum(case when month="Jul" then revenue else null end) as Jul\_Revenue,

    sum(case when month="Aug" then revenue else null end) as Aug\_Revenue,

    sum(case when month="Sep" then revenue else null end) as Sep\_Revenue,

    sum(case when month="Oct" then revenue else null end) as Oct\_Revenue,

    sum(case when month="Nov" then revenue else null end) as Nov\_Revenue,

    sum(case when month="Dec" then revenue else null end) as Dec\_Revenue

from Department

group by id

1193

select date\_format(trans\_date,"%Y-%m") as month, country, count(\*) as trans\_count, sum(state="approved") as approved\_count, sum(amount) as trans\_total\_amount, sum(case when state="approved" then amount else 0 end) as approved\_total\_amount

from transactions

group by country, date\_format(trans\_date,"%Y-%m")

1194

select group\_id,player\_id

from(

select player\_id,group\_id,score,dense\_rank() over(partition by group\_id order by score desc,p.player\_id) as rnk

from players p

left join (

select first\_player as player, sum(first\_score) as score

from(

select first\_player,first\_score from matches

union all

select second\_player,second\_score from matches) t

group by first\_player) t1

on p.player\_id=t1.player) t2

where rnk=1

1024

select person\_name

from(

select person\_name,sum(weight) over(order by turn) as sum\_weight

from queue) t

where sum\_weight<=1000

order by sum\_weight desc limit 1

1205

select date\_format(trans\_date,"%Y-%m") as month, country, sum(state="approved") as approved\_count,sum(case when state="approved" then amount else 0 end) as approved\_amount, sum(state="cb") as chargeback\_count, sum(case when state="cb" then amount else 0 end) as chargeback\_amount

from(

select \* from Transactions

union all

select trans\_id,country,"cb" as state,amount,c.trans\_date

from Chargebacks c

left join transactions t

on c.trans\_id=t.id)a t1

group by country, month

having approved\_amount or chargeback\_amount

1211

select query\_name, round(sum(rating/position)/count(\*),2) as quality, round(100\*sum(rating<3)/count(\*),2) as poor\_query\_percentage

from Queries

group by query\_name

1212

select t1.team\_id,t1.team\_name,ifnull(num\_points,0) as num\_points

from teams t1

left join(

select host\_team as team\_id,

    sum(case when host\_goals>guest\_goals then 3

        when host\_goals=guest\_goals then 1

        else 0 end) as num\_points

from(

select host\_team,guest\_team,host\_goals,guest\_goals from matches

union all

select guest\_team,host\_team,guest\_goals,host\_goals from matches) t

group by host\_team) t2

on t1.team\_id=t2.team\_id

order by num\_points desc, t1.team\_id

1225

select stat as period\_state, min(date) as start\_date,max(date) as end\_date

from(

select \*,date\_sub(date,interval rnk day) as new\_date

from(·

select \*,dense\_rank() over(partition by stat order by date) as rnk

from(

select "failed" as stat, fail\_date as date from failed where year(fail\_date)=2019

union all

select "succeeded" as stat, success\_date as date from succeeded where year(success\_date)=2019) t) t1)t2

group by stat,new\_date

order by date

先把success和fail两个表union all，按照stat进行row\_number（）排序，按照

Stat和Datesub（date，row\_number的结果）group by；聚合的最小日期为start date，最大日期为end date，在按照start date进行排序

1241

select post\_id,ifnull(num,0) as number\_of\_comments

from

(select distinct sub\_id as post\_id from submissions where parent\_id is null) t3

left join(

select parent\_id,count(\*) as num

from(

select distinct sub\_id,parent\_id

from submissions

where parent\_id is not null) t

group by parent\_id) t2

on t3.post\_id=t2.parent\_id

order by post\_id

1251

select p.product\_id,round(sum(price\*units)/sum(units),2) as average\_price

from prices p

join UnitsSold u

on p.product\_id=u.product\_id and u.purchase\_date between p.start\_date and p.end\_date

group by p.product\_id

1264

select distinct page\_id as recommended\_page

from likes

where user\_id in (

select distinct user2\_id

from(

select user1\_id,user2\_id from friendship

union all

select user2\_id,user1\_id from friendship) f

where user1\_id=1) and page\_id not in (select page\_id from likes where user\_id=1)

1270

select employee\_id

from Employees

where employee\_id<>1 and (manager\_id=1 or manager\_id in (select employee\_id from Employees where manager\_id=1) or manager\_id in (select employee\_id from Employees where manager\_id=1 or manager\_id in(select employee\_id from Employees where manager\_id=1)))

1280

select s.student\_id,s.student\_name,s1.subject\_name,ifnull(count(e.subject\_name),0) as attended\_exams

from students s

join subjects s1

left join examinations e on s.student\_id=e.student\_id and s1.subject\_name=e.subject\_name

group by s.student\_name,s1.subject\_name

order by s.student\_id,s1.subject\_name

1285

select min(log\_id) as start\_id,max(log\_id) as end\_id

from(

select log\_id,log\_id-row\_number() over(order by log\_id) as diff

from logs) t

group by diff

1294

select country\_name,case when temp<=15 then "Cold" when temp>=25 then "Hot" else "Warm" end as weather\_type

from countries c

join(

select country\_id,avg(weather\_state) as temp

from weather

where day between '2019-11-01' and '2019-11-30'

group by country\_id) t

on c.country\_id=t.country\_id

1303

select employee\_id,team\_size

from Employee e

join

(select team\_id,count(\*) as team\_size

from Employee

group by team\_id) t

on t.team\_id=e.team\_id

1308

select gender,day,sum(score\_points) over(partition by gender order by day) as total

from scores

order by gender,day

1321

select visited\_on,amount,round(amount/7,2) as average\_amount

from(

select visited\_on,sum(amount) over(order by visited\_on rows 6 preceding) as amount, row\_number() over(order by visited\_on) as rnk

from (select customer\_id,name,visited\_on,sum(amount) as amount from customer group by visited\_on) t

group by visited\_on) t1

where rnk>=7

注意2019-01-10有两个记录要先将表按时间聚合取每日sum(amount)在用窗口函数，用row number把前6天的数据筛掉

1322

select ad\_id,ifnull(round(clicked/(clicked+viewed)\*100,2),0) as ctr

from(

select ad\_id, sum(case when action='Clicked' then 1 else 0 end) as clicked,sum(case when action='Viewed' then 1 else 0 end) as viewed

from ads

group by ad\_id) t

order by ctr desc,ad\_id

1327

select product\_name,sum(unit) as unit

from products p

join(

select \*

from orders

where order\_date between '2020-02-01' and '2020-02-29') o

on p.product\_id=o.product\_id

group by p.product\_id

having sum(unit) >=100

1336

select con1, sum(con2 is not null) as cnt

from(

select user\_id,visit\_date,concat(user\_id,"-",visit\_date) as con1 from visits) v

left join

(select user\_id,transaction\_date,concat(user\_id,"-",transaction\_date) as con2 from transactions) t

on v.con1=t.con2

group by con1

1341

select (select u.name

from Users u

join(

select user\_id,count(movie\_id) as num

from Movie\_Rating

group by user\_id

order by count(movie\_id) desc, user\_id) t1

on t1.user\_id=u.user\_id

order by num desc, u.name limit 1) as results

union

select (select m.title

from Movies m

join(

select movie\_id,avg(rating) as rating

from Movie\_Rating

where created\_at between '2020-02-01' and '2020-02-29'

group by movie\_id

order by avg(rating) desc,movie\_id) t2

on m.movie\_id=t2.movie\_id

order by rating desc, m.title limit 1) as results

1350

select id,name

from Students

where department\_id not in (select id from Departments)

1355

select activity

from friends

group by activity

having count(\*) <> (select count(id) from friends group by activity order by count(id) desc limit 1) and count(\*) <> (select count(id) from friends group by activity order by count(id)  limit 1)

1364

select invoice\_id,customer\_name,price,contacts\_cnt,trusted\_contacts\_cnt

from invoices i

join(

select c1.customer\_name,c1.customer\_id,count(c2.user\_id) as contacts\_cnt, sum(case when c2.contact\_name in (select customer\_name from customers) then 1 else 0 end) as trusted\_contacts\_cnt

from customers c1

left join contacts c2 on c1.customer\_id=c2.user\_id

group by c1.customer\_id) t

on i.user\_id=t.customer\_id

order by invoice\_id

1369

select username,activity,startDate,endDate

from(

select \*,row\_number() over(partition by username order by startDate desc) rnk, count(\*) over(partition by username) cnt

from useractivity) t

where cnt=1 or rnk=2

1378

select unique\_id,e1.name

from Employees e1

left join EmployeeUNI e2

on e1.id=e2.id

1384

select \*

from(

select s.product\_id,product\_name,'2018' as report\_year, (case when period\_start <= '2018-01-01' and period\_end <='2018-12-31' then average\_daily\_sales\*(datediff(period\_end,'2018-01-01')+1) when period\_start <= '2018-01-01' and period\_end >= '2018-12-31' then average\_daily\_sales\*(datediff('2018-12-31','2018-01-01')+1)

when period\_start>='2018-01-01' and period\_end <='2018-12-31' then average\_daily\_sales\*(datediff(period\_end,period\_start)+1) when period\_start >= '2018-01-01' and period\_end>='2018-12-31' then average\_daily\_sales\*(datediff('2018-12-31',period\_start)+1) end) as total\_amount

from sales s

join product p

on s.product\_id=p.product\_id

where year(period\_start)<=2018 and year(period\_end)>=2018

group by product\_id

union all

select s.product\_id,product\_name,'2019' as report\_year, (case when period\_start <= '2019-01-01' and period\_end <='2019-12-31' then average\_daily\_sales\*(datediff(period\_end,'2019-01-01')+1) when period\_start <= '2019-01-01' and period\_end >= '2019-12-31' then average\_daily\_sales\*(datediff('2019-12-31','2019-01-01')+1)

when period\_start>='2019-01-01' and period\_end <='2019-12-31' then average\_daily\_sales\*(datediff(period\_end,period\_start)+1) when period\_start >= '2019-01-01' and period\_end>='2019-12-31' then average\_daily\_sales\*(datediff('2019-12-31',period\_start)+1) end) as total\_amount

from sales s

join product p

on s.product\_id=p.product\_id

where year(period\_start)<=2019 and year(period\_end)>=2019

group by product\_id

union all

select s.product\_id,product\_name,'2020' as report\_year, (case when period\_start <= '2020-01-01' and period\_end <='2020-12-31' then average\_daily\_sales\*(datediff(period\_end,'2020-01-01')+1) when period\_start <= '2020-01-01' and period\_end >= '2020-12-31' then average\_daily\_sales\*(datediff('2020-12-31','2020-01-01')+1)

when period\_start>='2020-01-01' and period\_end <='2020-12-31' then average\_daily\_sales\*(datediff(period\_end,period\_start)+1) when period\_start >= '2020-01-01' and period\_end>='2020-12-31' then average\_daily\_sales\*(datediff('2020-12-31',period\_start)+1) end) as total\_amount

from sales s

join product p

on s.product\_id=p.product\_id

where year(period\_start)<=2020 and year(period\_end)>=2020

group by product\_id) t

order by product\_id,report\_year

1393

select stock\_name,

sum(case when operation="Buy" then -price else price end) as capital\_gain\_loss

from Stocks

group by stock\_name

1398

select t.customer\_id,customer\_name

from(

select customer\_id,sum(case when product\_name='A' then 1 else 0 end) as a\_cnt,

sum(case when product\_name='B' then 1 else 0 end) as b\_cnt,sum(case when product\_name='C' then 1 else 0 end) as c\_cnt

from orders

group by customer\_id) t

join customers c

on t.customer\_id=c.customer\_id

where a\_cnt>=1 and b\_cnt>=1 and c\_cnt=0

1407

select name,ifnull(travelled\_distance,0) as travelled\_distance

from users u

left join(

select user\_id,sum(distance) as travelled\_distance

from rides

group by user\_id) t

on u.id=t.user\_id

order by travelled\_distance desc, name

1412

select student\_id,student\_name

from student

where student\_id in(

select distinct student\_id

from exam

where student\_id not in (

select distinct student\_id

from (

    select \*,dense\_rank() over (partition by exam\_id order by score desc) as rnk,dense\_rank() over (partition by exam\_id order by score) as rnk2

    from exam) t

where rnk=1 or rnk2=1))

用dense rank排序而不用row number：如果存在一门考试全部人成绩都一样row number是能取第一个人和第二个人忽略了中间的人

1241

select q.id,q.year,ifnull(npv,0) as npv

from Queries q

left join NPV n

on q.id=n.id and q.year=n.year

1435

select "[0-5>" as "bin", sum(case when duration/60 < 5 then 1 else 0 end) as total

from sessions

union all

select "[5-10>" as "bin", sum(case when duration/60 >= 5 and duration/60 <10 then 1 else 0 end) as total

from sessions

union all

select "[10-15>" as "bin", sum(case when duration/60 >= 10 and duration/60 <15 then 1 else 0 end) as total

from sessions

union all

select "15 or more" as "bin", sum(case when duration/60 >=15 then 1 else 0 end) as total

from sessions

1440

select left\_operand,operator,right\_operand, case when operator='>' and v1>v2 then 'true' when operator='=' and v1=v2 then 'true' when

operator='<' and v1<v2 then 'true' else 'false' end as 'value'

from(

select left\_operand,operator,right\_operand,v1.value as v1,v2.value as v2

from expressions e

left join variables v1 on e.left\_operand=v1.name

left join variables v2 on e.right\_operand=v2.name) t

1445

select sale\_date, sum(case when fruit="apples" then sold\_num

                when fruit="oranges" then -sold\_num end) as diff

from sales

group by sale\_date

1454

select \*

from accounts

where id in (

select id

from(

select id,login\_date,i\_day,count(i\_day) as cnt

from(

select id, login\_date, date\_sub(login\_date,interval rnk day) as i\_day

from(

select distinct id,login\_date,dense\_rank() over(partition by id order by login\_date) as rnk from logins) t

) t1

group by id,i\_day) t2

where cnt>=5)

order by id

1468

select company\_id,employee\_id,employee\_name,

    round((case when max\_salary<1000 then salary

        when max\_salary>=1000 and max\_salary<10000 then salary\*0.76

        else salary\*0.51 end),0) as salary

from(

select \*, max(salary) over(partition by company\_id) as max\_salary

from salaries) s

1479

select item\_category as category,ifnull(sum(Monday),0) as Monday,ifnull(sum(Tuesday),0) as Tuesday,ifnull(sum(Wednesday),0) as Wednesday ,ifnull(sum(Thursday),0) as Thursday,ifnull(sum(Friday),0) as Friday ,ifnull(sum(Saturday),0) as Saturday,ifnull(sum(Sunday),0) as Sunday

from items i

left join (

select item\_id,

    sum(case when weekday(order\_date)=0 then quantity else 0 end) as "Monday",

    sum(case when weekday(order\_date)=1 then quantity else 0 end) as "Tuesday",

    sum(case when weekday(order\_date)=2 then quantity else 0 end) as "Wednesday",

    sum(case when weekday(order\_date)=3 then quantity else 0 end) as "Thursday",

    sum(case when weekday(order\_date)=4 then quantity else 0 end) as "Friday",

    sum(case when weekday(order\_date)=5 then quantity else 0 end) as "Saturday",

    sum(case when weekday(order\_date)=6 then quantity else 0 end) as "Sunday"

from orders

group by item\_id) o

on i.item\_id=o.item\_id

group by item\_category

order by item\_category

1485

select sell\_date, count(distinct product) as num\_sold, **group\_concat**(distinct product) as products

from Activities

group by sell\_date

1495

select title

from content

where kids\_content="Y" and content\_type="Movies" and content\_id in (select content\_id from TVProgram where program\_date between '2020-06-01' and '2020-06-30')

1501

select c1.name country

from(

select caller\_id,duration from Calls

union all

select callee\_id,duration from Calls) c

join Person p on p.id=c.caller\_id

join country c1 on left(p.phone\_number,3)=c1.country\_code

group by c1.name

having avg(duration) > (select avg(duration) from calls)

1511

select t1.customer\_id,name

from (

select c.customer\_id,name,sum(case when order\_date between '2020-06-01' and '2020-06-30' then quantity\*price end) as june, sum(case when order\_date between '2020-07-01' and '2020-07-31' then quantity\*price end) as july

from (

select \*

from orders

where order\_date between '2020-06-01' and '2020-07-31') t

join product p

on t.product\_id=p.product\_id

join customers c

on c.customer\_id=t.customer\_id

group by c.customer\_id) t1

where june>=100 and july>=100

**1517**

select \*

from users

where mail regexp '^[A-Za-z]+[a-zA-Z0-9\_\\.\\-]{0,}@leetcode.com'

1527

select \*

from Patients

where conditions like '%DIAB1%'

1532

select c.name as customer\_name,c.customer\_id,t.order\_id,t.order\_date

from

(select customer\_id,order\_id,order\_date,row\_number() over(partition by customer\_id order by order\_date desc) as od from Orders) t

join customers c

on c.customer\_id=t.customer\_id

where od <=3

order by name,c.customer\_id,t.order\_date desc

1543

select trim(lower(product\_name)) as product\_name, date\_format(sale\_date,"%Y-%m") as sale\_date,

count(sale\_id) as total

from sales

group by trim(lower(product\_name)),date\_format(sale\_date,"%Y-%m")

order by product\_name,sale\_date

1549

select product\_name,p.product\_id,order\_id,order\_date

from orders o

join products p

on o.product\_id=p.product\_id

where (p.product\_id,order\_date) in (

select product\_id,max(order\_date)

from orders

group by product\_id)

order by product\_name,product\_id,order\_id

1555

select user\_id,user\_name, credit+ifnull(amount,0) as credit, if(credit+ifnull(amount,0)>0,"No","Yes") as credit\_limit\_breached

from users u

left join(

select paid\_by as id, sum(t.amount) as amount

from(

select paid\_by,paid\_to,-amount as amount from Transactions

union all

select paid\_to,paid\_by,amount as amount from Transactions

) t

group by paid\_by) t1

on t1.id=u.user\_id

1565

select date\_format(order\_date,"%Y-%m") as month,

        count(distinct order\_id) as order\_count,count(distinct customer\_id) as customer\_count

from (select \* from orders where invoice >20) o

group by month

1571

select w.name as warehouse\_name, sum(volume\*units) as volume

from warehouse w

left join(

select \*,width\*length\*height as volume from products) t on t.product\_id=w.product\_id

group by w.name

1581

select customer\_id,count(visit\_id) as count\_no\_trans

from(

select v.visit\_id,v.customer\_id,t.transaction\_id

from visits v

left join transactions t

on v.visit\_id=t.visit\_id

where transaction\_id is null) t1

group by customer\_id

1587

select u.name,balance

from users u

join (

select account,sum(amount) as balance

from transactions

group by account) b

on u.account=b.account

where balance>10000

1596

select distinct customer\_id,p.product\_id,product\_name

from products p

join(

select customer\_id,product\_id,dense\_rank() over(partition by customer\_id order by cnt desc) as rnk

from(

select customer\_id,product\_id,count(product\_id) as cnt

from orders

group by customer\_id,product\_id

order by customer\_id,product\_id) t) t1

on t1.product\_id=p.product\_id

where rnk=1

order by customer\_id,product\_id

1607

select seller\_name

from seller

where seller\_id not in (select seller\_id from orders where sale\_date between '2020-01-01' and '2020-12-31')

order by seller\_name

1623

select a.student\_name as member\_A,b.student\_name as member\_B,c.student\_name as member\_C

from SchoolA a

join SchoolB b on a.student\_id <> b.student\_id and a.student\_name <> b.student\_name

join SchoolC c on a.student\_id <> c.student\_id and b.student\_id <> c.student\_id and b.student\_name<>c.student\_name and a.student\_name<>c.student\_name

1633

select contest\_id,round(count(distinct user\_id)/(select count(\*) from users)\*100,2) as percentage

from register

group by contest\_id

order by percentage desc, contest\_id

1661

select distinct machine\_id, round(avg(time\_interval) over(partition by machine\_id),3) as processing\_time

from (

select machine\_id, process\_id, sum(case when activity\_type = "start" then -timestamp else timestamp end) as time\_interval

from activity

group by machine\_id,process\_id) t

1667

select user\_id,concat(upper(left(name,1)),lower(right(name,length(name)-1))) as name

from Users

order by user\_id

1677

select name, rest,paid,canceled,refunded

from(

select product\_id,sum(rest) as rest,sum(paid) as paid, sum(canceled) as canceled, sum(refunded) as refunded

from invoice

group by product\_id) t

join product p

on t.product\_id=p.product\_id

order by name

1683

select tweet\_id

from tweets

where length(content) >15

1693

select date\_id,make\_name,count(distinct lead\_id) as unique\_leads,count(distinct partner\_id) as unique\_partners

from DailySales

group by date\_id,make\_name

1699

select from\_id as person1, to\_id as person2, count(\*) as call\_count,sum(duration) as total\_duration

from(

select from\_id,to\_id,duration from calls

union all

select to\_id,from\_id,duration from calls) t

group by from\_id,to\_id

having from\_id < to\_id